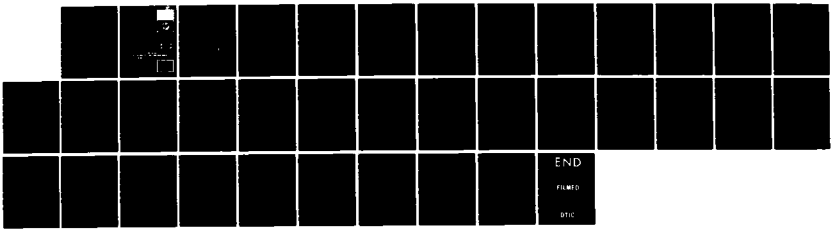
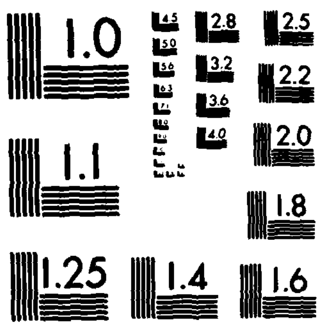
AD-A150 613 ON SEQUENCING RETRIEVALS IN AN AUTOMATED STORAGE/RETRIEVAL SYSTEM(U) GEORGIA INST OF TECH ATLANTA PRODUCTION AND DISTRIBUTION RESE.. 1/1

UNCLASSIFIED M H HAN ET AL. OCT 84 PDRC-84-06 F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD-A150 613

Georgia Institute of Technology

ON SEQUENCING RETRIEVALS IN AN AUTOMATED STORAGE/RETRIEVAL SYSTEM

by

Min-Hong Han
Leon F. McGinnis
Jin Shen Shieh
John A. White

PDRC 84-06

PRODUCTION and DISTRIBUTION RESEARCH CENTER

SCHOOL OF INDUSTRIAL
AND SYSTEMS
ENGINEERING
GEORGIA INSTITUTE OF TECHNOLOGY
A UNIT OF THE UNIVERSITY SYSTEM
OF GEORGIA
ATLANTA, GEORGIA 30332

PDRC Report Series 84-06
October 1984

ON SEQUENCING RETRIEVALS IN AN AUTOMATED STORAGE/RETRIEVAL SYSTEM

by

Min-Hong Han
Leon F. McGinnis
Jin Shen Shieh
John A. White

PDRC 84-06

School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, Georgia 30332

This work was supported by the Office of Naval Research under Contract No. N00014-83-K-0147. Reproduction is permitted in whole or in part for any purpose of the U. S. Government.

On Sequencing Retrievals in an Automated Storage/Retrieval System

Abstract

The problem addressed in this paper is the possible improvement in AS/RS throughput by sequencing retrievals in dual command cycles. An expected cycle time model is developed for a nearest-neighbor sequencing heuristic along with a lower bound on average cycle time. Some critical elements in evaluating retrieval sequencing heuristics are also highlighted.

Introduction

Automated storage/retrieval systems (AS/RS) are widely used in warehousing, and often found in manufacturing. The basic components of an AS/RS are the storage racks and the S/R machines. The corresponding design parameters are storage capacity, or number of storage locations, and throughput capacity, or maximum number of transactions per hour.

For a single aisle in an AS/RS, the throughput capacity is defined by the inverse of the average cycle time, i.e., the average amount of time required for the S/R machine to store and/or retrieve a unit load. Cycle time includes travel time and shuttle time, which is the time required to pick up or deposit a load, and typically depends on the S/R specification. Average travel time, on the other hand, depends on both the speed of the S/R machine and the dimensions of the rack.

Estimating the average S/R cycle time is a fundamental step in AS/RS design. One way to improve throughput capacity of a storage aisle is to reduce the dimensions of the rack, thereby reducing average travel time. In order to maintain a constant storage capacity for the overall system as the number of storage locations per aisle is reduced, additional aisles will be required. Since the cost of the total system is very sensitive to the number of aisles, it is critical to know when the average S/R cycle time is small enough to satisfy the throughput requirement for a storage aisle.

The problem of estimating S/R cycle time in a conventional unit load AS/RS has been studied by Gudehus [6], Graves, Hausman and Schwartz [5, 7, and 13], Bozer and White [1], and Rizo-Patron, Bozer, and McGinnis [12]. In these studies, two types of S/R machine cycles are examined:

single command (SC):

> a single storage or retrieval is performed; storage cycle time is equal to the sum of load (pick-up) time plus travel time to storage location plus unload (deposit) time plus return time to P/D station; retrieval cycle time is similar; and

dual command (DC):

> both a storage and a retrieval are performed; cycle time is the sum of load time plus travel time to storage location plus unload plus travel time to retrieval point plus load time plus return time to P/D station plus unload time.

A primary objective in each of the studies is the development of analytic expressions for the expected time to complete a cycle of specified type. A common practice, reflected in the studies, is that both storage and retrieval requests are processed in first-come-first-served (FCFS) manner, and that storages go to the closest open location (COL). The FCFS assumption is reasonable for storages, since most AS/R systems are interfaced with a conveyor loop for input and output. In this case, there is no capability for changing the order of loads presented for storages. However, for retrievals, the FCFS assumption is less compelling. Retrieval requests are nothing more than messages, typically in some control computer. Hence, there is no intrinsic reason why they cannot be rearranged into any convenient sequence.

The available expected cycle time equations can be quite accurate, as demonstrated by the detailed simulation results given by Rizo-Patron, Bozer and McGinnis [12]. A natural question, however, is whether or not the AS/RS throughput performance can be improved by cleverly sequencing the retrievals, so that the time spent in traveling between the storage and retrieval locations of dual command cycles is reduced. If so, the expected cycle time models currently available will overestimate travel time, and therefore underestimate throughput.

Sequencing retrieval requests optimally is a complex problem. In the first place, the list of retrievals changes through time as old requests are filled and new requests appear. How should the sequencing of retrievals be managed in such a dynamic situation? Two alternatives exist: select a "block" of retrievals, sequence the block, and repeat; or resequence the list every time a new request is added, but employ due dates or priorities to ensure that a retrieval at the far end of the aisle is not excessively delayed. In this paper, we adopt the first alternative. While it may not be the best, it is simple to implement and provides a lower bound on throughput improvement, i.e., more sophisticated strategies may perform even better.

The problem of optimally sequencing a given list of retrievals is provably hard. In the case of a single open location for the initial storage operation, the sequencing problem is equivalent to a traveling salesman problem [4,8,9]. Thus, it is NP-complete [2,10,11]. Further, the problem is no easier to solve if there exist multiple open locations.

Optimum sequencing of retrievals is known to be, at least theoretically, a difficult problem. Also, implementation of retrieval sequencing requires modification of existing control software. Therefore, the first question to be answered is:

"Can retrieval sequencing make a significant improvement?"

To answer the question, consider the relative increase in throughput resulting from a given relative reduction in travel time between storage point and retrieval point (referred to as travel-between). Analysis indicates for a typical AS/RS configuration that operating with 100% dual commands, a 60% reduction in travel-between times yields 12% increase in throughput. The latter is considered to be significant.

The next question to be answered is:

"Can a 60% reduction in travel-between be achieved with reasonable effort?"

To answer the question, a "nearest neighbor" heuristic is applied in sequencing retrievals. The heuristic is easy to implement and requires minimal computational resources. In fact, all computations were performed using compiled Basic language and an IBM PC-XT microcomputer.

The results obtained indicate that a block of size 15 to 20 retrievals reduces travel-between by 60%, with one open location provided. With more open locations, greater reduction results. A lower bound on average dual cycle time can be established for different block sizes and numbers of open locations. Based on the lower bound, the opportunity for further improvement in throughput can be estimated.

Modeling Throughput Improvements

Suppose that only dual commands will be performed and that there is a retrieval list, which can be as large as desired. The travel-between for a series of dual commands is illustrated in Figure 1 for a single open location and eight retrieval points. (In practice, there often will be several open locations, but that would needlessly clutter the figure.) The dashed line in the figure represents the travel-between using FCFS for retrievals, while the solid line represents the travel-between using a "nearest-neighbor" heuristic for sequencing retrievals. In the example the travel-between is much smaller for nearest-neighbor sequencing.

Figure 1 here

The expected travel time using FCFS for single and dual commands is modeled in [1] as follows:

s_h = horizontal travel speed

s_v = vertical travel speed

L = rack length

H = rack height

$$t_h = \frac{L}{s_h}$$

$$t_v = \frac{H}{s_v}$$

$$T = \max(t_h, t_v)$$

$$E(SC) = [1 + \frac{1}{3} b^2]T \tag{1}$$

= expected single command travel time.

$$E(TB) = [\frac{1}{3} + \frac{1}{6} b^2 - \frac{1}{30} b^3]T \tag{2}$$

= expected travel-between time for DC.

$$E(DC) = [\frac{4}{3} + \frac{1}{2} b^2 - \frac{1}{30} b^3]T \tag{3}$$

= expected dual command travel time.

The expression for expected travel time can be modified in a simple way to account for rack utilization, as shown in [12].

Let τ represent the load/unload time and suppose that a set of n dual commands is to be performed. The total time to perform the 2n operations (n storages plus n retrievals) in FCFS or random order is given as $[nE(DC) + 4n\tau]$. Therefore, the average throughput per unit time, γ, is:

$$\gamma = \frac{2n}{nE(DC) + 4n\tau} \tag{4}$$

In performing the n dual commands, the expected total travel-between is given by nE(TB).

Now suppose that an α100% reduction in average travel-between can be achieved through retrieval sequencing. What improvement in throughput will result? The new throughput, γ', will be;

$$\gamma' = \frac{2n}{n[E(DC) - \alpha E(TB)] + 4n\tau}$$

$$= \frac{2}{E(DC) - \alpha E(TB) + 4\tau} \tag{5}$$

Let $\gamma' = \gamma(1+\beta)$ so that $\beta100\%$ is the resulting improvement in throughput. Solving for β in terms of α,

$$\beta = \frac{E(DC) + 4\tau}{E(DC) - \alpha E(TB) + 4\tau} - 1 \qquad (6)$$

If travel-between is completely eliminated in a typical AS/RS, with $b=1$, $T=1$ minute, and $\tau=0.2$ minutes, the throughput improvement will be 22%, as calculated below:

$$\beta = \frac{1.8 + 4(0.2)}{1.8 - 0.47 + 4(0.2)} - 1 = 0.22$$

Figure 2 illustrates the relationship between β and α, for the case of $\tau = 0.2$, $T = 1$, and $b = 0.6$, 0.8, and 1.0. The relationship between β and α is essentially unchanged for other reasonable values of τ and T.

Figure 2 here

The results in Figure 2 lead to the conclusion:

> For significant improvement in throughput (i.e., 10 to 15%), the travel-between must be reduced by over 50%, relative to FCFS retrieval.

Thus, a "threshold" performance criterion is established for any retrieval sequencing algorithm to be a serious candidate for implementation.

Analysis of Nearest-Neighbor Heuristic

The nearest-neighbor heuristic is a very simple procedure for sequencing retrievals. Let R be the block of n retrieval locations and denote by S the set of initial storage locations, i.e., open locations.

Then the nearest-neighbor heuristic is;

```
While $R \neq \emptyset$
    1. Select a pair, $r \in R$ and $s \in S$, with minimum travel-between
       distance.
    2. Perform a DC, storing in s and retrieving from r.
    3. $R \leftarrow R - \{r\}$
    4. $S \leftarrow S - \{s\} + \{r\}$
End
```

It is reasonable to expect that the average travel-between will decrease as the size of the retrieval block increases and as the number of open locations increases. In the following, an analytic model is developed for the expected value of $TB^{NN}_{n,m}$, i.e., the travel-between using the nearest neighbor heuristic when n retrievals and m open locations are given. The model is then evaluated using Monte Carlo sampling.

Bozer and White [1] show that the distance between two randomly selected locations is a random variable, Z, with pdf and cdf:

$$f(z) = \begin{cases} (2 - 2z)\left[2\left(\frac{z}{b}\right) - \left(\frac{z}{b}\right)^2\right] & 0 < z < b \\ \quad + (2z - z^2)\left[\frac{2}{b} - \frac{2z}{b^2}\right] & \\ 2(1 - z) & b < z < 1 \end{cases} \tag{7}$$

$$F(z) = \begin{cases} (2z - z^2)\left[2\left(\frac{z}{b}\right) - \left(\frac{z}{b}\right)^2\right] & 0 < z < b \\ 2z - z^2 & b < z < 1 \end{cases} \tag{8}$$

Given a sample of n random distances, the smallest of them, z_n, is a random variable having density function:

$$g(z_n) = n[1 - F(z_n)]^{n-1} f(z_n) \quad 0 < z_n < 1 \tag{9}$$

Therefore, the expected value, $E(Z_n)$ can be found from:

$$E(Z_n) = \int_0^1 z\, n[1 - F(z)]^{n-1} f(z)dz \tag{10}$$

using the expressions in (7) and (8) above for $f(z)$ and $F(z)$. We have evaluated (10) numerically for $n = 1,\dots,40$, and for several values of b. The results are displayed in Table 1.

Table 1 here

The numerical results obtained will be used to approximate the average value of $TB^{NN}_{n,m}$ via the following argument. If the complete sequence of storage and retrieval operations is observed, the last dual command will have a specified retrieval location and a choice among m open locations for its accompanying storage location. The next to the last dual command will involve travel between the particular retrieval location and one of m currently open locations or the last remaining retrieval point (should it be performed first). Continuing the process, the first dual command will have travel-between corresponding to one retrieval location and a choice of one of m open locations or n-1 retrieval points.

Therefore, the expected value of $TB_{n,m}$ for nearest-neighbor sequencing, $E(TB^{NN}_{n,m})$, can be approximated as:

$$\hat{E}(TB^{NN}_{n,m}) = \frac{1}{n} \sum_{i=m}^{n+m-1} E(Z_i) \qquad (11)$$

Then the expected dual command cycle time for the nearest-neighbor heuristic $E(DC^{NN}_{n,m})$, can be approximated as:

$$\hat{E}(DC^{NN}_{n,m}) = E(SC) + E(TB^{NN}_{n,m}) \qquad (12)$$

Using the values of $E(Z_n)$ from Table 1, the values of $\hat{E}(TB^{NN}_{n,m})$ and of $\hat{E}(DC^{NN}_{n,m})$ are tabulated in Table 2, for $b = 1$ and $T = 1$. Note that $\hat{E}(TB^{NN}_{n,m})$ is not symmetric with respect to n and m.

Table 2 here

To assess the accuracy of the approximation in (11), and consequently in (12), the nearest-neighbor heuristic was applied to retrieval lists generated by Monte Carlo sampling. The results are shown in Table 3. In the Monte Carlo sampling procedure, points were randomly chosen in the continuous rectangle with dimensions $1 \times b$. For each sample, n random retrieval points and m random open points were generated. The nearest-neighbor heuristic rule was applied until the n retrievals were complete. For each n and m combination a total of 1,000 samples were generated. The values of $\hat{E}(DC^{NN}_{n,m})$ from approximation equation (12) are plotted together with the results of Monte Carlo sampling, in Figure 3.

Table 3 here

Figure 3 here

The approximation results appear to be quite accurate over practical ranges of n and m. Based on the approximation, it appears that the nearest-neighbor heuristic will yield at least a 60% reduction in travel-between with a block size of 15 to 20 (refer to Table 2). Based on the expected value model, Figure 4 illustrates the impact of block size on throughput increase for varying number of open locations. As the number of open locations increases, the throughput improvement increases, and is relatively insensitive to block size. For example, with three open locations, a 15% increase in throughput is achieved with a block size of 4. Whereas, the same improvement with a single open location requires a block size of 11.

Figure 4 here

Lower Bound on Expected Dual Cycle Time

The approximation in (12) provides an upper bound on the best possible average dual command cycle time. Furthermore, the corresponding nearest-neighbor heuristic can provide a significantly smaller average cycle time than the FCFS rule.

A question that remains is:

> "How much additional reduction in average cycle time is theoretically possible?"

To answer the question, a lower bound on the average dual command cycle time is needed. Note that E(SC) provides a lower bound. Unfortunately, it is not a very tight bound.

To establish a better bound, the notions of order statistics are used again. In addition, it is desirable to distinguish "travel-between" and "effective travel-between". To illustrate, Figure 5 shows a single retrieval point, denoted r, which is to be matched with some storage point to form a dual command cycle. If the storage point, e.g., point s, lies outside the crosshatched area the cycle time will be $T_s + T_{sr} + T_r$, with T_{sr} the travel-between. If the storage point, e.g., point t, lies inside the crosshatched area, the cycle time will be $T_t + T_{tr} + T_r$.

Figure 5 here

Because the S/R machine moves simultaneously in the horizontal and vertical directions, $T_t + T_{tr} = T_r$. In other words, the dual command cycle with storage at t and retrieval at r has exactly the same travel time as a single command retrieval from r. The crosshatched area is referred to as the "no cost zone", since a storage operation within the region is essentially free with regard to travel time. Also, in this case, the dual command cycle is said to have zero effective travel-between.

Now suppose that there are n randomly distributed retrieval points and m randomly distributed open locations. Denote by r (s) an arbitrary retrieval (storage) point and by T_r (T_s) the travel time between r (s) and the P/D station. Let T_{rs} be the travel time between r and s. If s is in the no cost zone for r, the total travel time is $2T_r$, but if s is outside the no cost zone for r, the total travel time is $T_s + T_{sr} + T_r$. Let p_m denote the probability, given m randomly distributed open locations and a random retrieval point, that at least one open location falls in the no cost zone for the retrieval point.

Using the notion of a no cost zone, a lower bound can be established for dual command cycle time using the following argument. Suppose an optimal retrieval sequencing rule exists, with resulting expected values $E(DC^*_{n,m})$ and $E(TB^*_{n,m})$. Note that $E(TB^*_{n,m})$ is larger than the average effective travel-between, because it includes nonzero travel-between values for storages in the no cost zone. Under the assumption that storages and retrievals are always randomly distributed,

$$E(DC^*_{n,m}) = E(SC) + [1 - p_m]E(TB^*_{n,m}) \tag{13}$$

where the term, $[1-P_m]E(TB^* n,m)$ represents the average effective travel-between. It can be argued that $E(TB^*_{n,m}) > E(Z_{n+m-1})$ since Z_{n+m-1} is the random variable for the smallest possible travel-between. Therefore

$$E(DC^{LB}_{n,m}) = E(SC) + (1-p_m)E(Z_{n+m-1}) < E(DC^*_{n,m})$$

and any heuristic solution is greater than $E(DC^{LB}_{n,m})$, i.e.,

$$\hat{E}(DC^{NN}_{n,m}) > E(DC^*_{n,m}) > E(DC^{LB}_{n,m}) \tag{14}$$

The probability of having no open locations in the no cost zone can be estimated by the following reasoning. Under the assumption that open locations are randomly distributed in a 1xb rectangle, the number of open locations found in an area of size Δ will follow the Poisson distribution with parameter $\lambda\Delta$,

$$\text{where } \lambda = \frac{\text{number of open locations}}{\text{total area of rectangle}}$$

Therefore, the probability of having no open location in an area of size Δ will be:

$$\begin{aligned} \Pr(\text{no open location in area of size } \Delta) &= \exp(-\lambda\Delta) \\ &= \exp\left(-\frac{m}{1 \times b}\Delta\right) \\ &= \exp(-m\Delta) \qquad \text{for } b = 1 \end{aligned}$$

When b = 1, the expected value of the area of the no cost zone is 0.125. Consequently, the lower bound of dual cycle time is:

$$E\left(DC_{n,m}^{LB}\right) = E(SC) + E(Z_{n+m-1})\exp(-0.125\ m) \qquad (15)$$

The values of $E\left(DC_{n,m}^{LB}\right)$ are tabulated in Table 4 for different values of n and m. In Figure 6, the lower bound $E\left(DC_{n,m}^{LB}\right)$ is compared with $E\left(DC_{n,m}^{NN}\right)$ for n = 1,...,20 and m = 1 and 10. Recall that for this case, E(SC) = 1.333, so the proposed lower bound is a significant improvement.

Table 4 here

Figure 6 here

The final step in the analysis is to examine the results in terms of throughput. An upper bound on throughput can be determined using the lower bound on average dual command cycle time. Also, the expected throughput can be determined from the expected dual command cycle time using the nearest-neighbor sequencing heuristic. The gap between the two, or potential for further improvement is:

$$r = \frac{\dfrac{1}{E\left(DC_{n,m}^{LB}\right) + 4\tau}}{\dfrac{1}{E\left(DC_{n,m}^{NN}\right) + 4\tau}} - 1 \qquad (16)$$

Values of r for $\tau = 0$ are given in Table 5. Note that the gap never exceeds 8%. For realistic values of τ, say 0.2 minutes, the maximum gap is close to 5%. Thus it can be concluded for this particular application that the nearest-neighbor sequencing rule is quite effective. Its average performance is within 5 - 8% of optimum. As an aside, its computational requirements are quite modest — using compiled Basic on an IBM PC/XT, the time to generate and sequence a sample of 20 retrieval points was less than one second.

Table 5 here

An Improved Heuristic - Maybe

The analysis of the lower bound on the average dual command cycle time employs the notion of a no cost zone. Such an approach leads natually to the consideration of heuristics that exploit the possibility of zero effective travel-between. As an example, consider the shortest-leg heuristic, defined as follows:

```
Repeat
    For each $s \in S$
        $\tau_{sr(s)} = \min_{r \in R} [T_s , T_{sr}]$
    Endfor
    $\tau_{\sigma \ell} = \min_{s \in S} [\tau_{sr(s)}]$
    $S = S + \{\ell\} - \{\sigma\}$
    $R = R - \{\ell\}$
Until $R = \emptyset$
```

The shortest-leg heuristic selects the storage and retrieval points that require the least travel (shortest-leg) to the retrieval point. Observe that for the retrieval point selected, if there is a storage point with zero effective travel, then such a point will be selected.

Figure 7 displays Monte Carolo sampling results for the shortest-leg heuristic with $n < 9$ and $m = 3$ or 6. Each experimental value is the average of 1000 samples. Several conclusions can be drawn from the figure. First, the shortest-leg heuristic consistently outperformed the nearest-neighbor heuristic. Second, the sampling results for the nearest-neighbor agree very well with the predictions of the analytic model. Third, for large m and small n, the shortest-leg heuristic yields results quite close to the lower bound.

Figure 7 here

The natural conclusion, based on Figure 7, is that the shortest-leg heuristic would be preferred. Note, however, that the method used in Monte Carlo sampling restarted the system for each sample, i.e., generated both new retrieval points and new storage points. An alternate sampling regime is to resume after each sample, i.e., retain the open locations at the end of a sample for use in the next sample, and generate only the new retrieval locations.

Surprisingly, the resumed sampling results were dramatically different from those for restart sampling. Figure 8 presents the results for both sampling regimes, along with the analytic lower bound. The nearest-neighbor heuristic yields almost identical results for both sampling regimes. The shortest-leg results, however, deteriorate markedly under resumed sampling, and are, in fact, significantly inferior to the nearest-neighbor heuristic.

Figure 8 here

It appears that the shortest-leg heuristic has an undesirable side effect on the distribution of the open locations. The conjecture is at least partially verified by Figure 9, which shows for each sampling regime, the average distance from the P/D station to open locations after every 50 samples. For restarted sampling, the average value fluctuates around 0.667, as expected. On the other hand, with resumed sampling, the average distance increases to a value around 0.88. This is quite remarkable, since the maximum possible distance is only 1.0!

Figure 9 here

It is not so important whether or not the nearest-neighbor heuristic is the best. Rather, the point is that every conceivable heuristic will have its own distinctive dynamic behavior. Any procedure for evaluating the heuristics must take into account such dynamic behavior. In the case of the shortest-leg heuristic, the open locations, although randomly distributed to begin with, tend to migrate over time to undersirable locations far from the P/D station. In contrast, the nearest-neighbor heuristic tends to preserve the randomness of the open locations.

Conclusions

We have addressed the problem of improving throughput capacity of unit load AS/RS by sequencing the retrieval requests. The important results presented are:

(1) the conclusion that a 10 - 15% improvement in throughput can be obtained by reducing the average travel-between component of dual command cycle by 50% or more;

(2) an equation for approximating the average dual command cycle time using a nearest-neighbor sequencing heuristic:

(3) a lower bound on the average dual command cycle time for <u>any</u> block sequencing rule;

(4) the conclusion that the nearest-neighbor heuristic obtains average throughput within 5 - 8% of the maximum possible average throughput; and

(5) a demonstration of the importance of dynamic analysis of sequencing heuristics.

The results in (2) and (3) are, so far as we know, unique.

Naturally, the issues addressed are relevant only if the AS/RS is throughput bound, perhaps in peak transaction periods. In addition, two key assumptions are that the arrival processes for storages and retrievals are random, and that a randomized storage policy (i.e., no "zones") is used. These assumptions appear to be quite reasonable for many work-in-process applications as well as more traditional warehousing systems.

REFERENCES

1. Bozer, Y. A., and J. A. White (1984). "Travel Time Models for Automated Storage/Retrieval Systems," forthcoming in IIE Transactions.

2. Garey, M., and D. S. Johnson (1979). Computers and Intractability: A Guide to the Theory of Computing, Freeman Press, San Francisco, California.

3. Goetschalckx, M. (1983). Storage and Retrieval Policies for Efficient Order Picking Operations, unpublished Ph.D. dissertation, Georgia Institute of Technology.

4. Golden, B., L. Bodin, T. Doyle, and W. Stewart (1980). "Approximate Traveling Salesman Algorithms," Operations Research. Vol. 28, No. 3, pp. 694-711.

5. Graves, S. C., W. H. Hausman, and L. B. Schwarz (1977). "Storage Retrieval Interleaving in Automatic Warehousing Systems," Management Science, Vol. 23, No. 9, pp. 935-945.

6. Gudehus, T. (1973). Grundlagen der Kommisoniertechnik: Dynamik der Waren-Verteil und Lagersysteme, (Principles of order picking: Operations in Distribution and Warehouse Systems) W. Girardet, Essen, West Germany.

7. Hausman, W. H., L. B. Schwarz, and S. C. Graves (1976). "Optimal Storage Assignment in Automatic Warehousing Systems," Management Science, Vol. 22, No. 6, pp. 625-638.

8. Held, M. and R. M. Karp (1970). "The Traveling Salesman Problem and Minimum Spanning Trees," Operations Research, Vol. 18, No. 6, pp. 1138-1162.

9. Held, M. and R. M. Karp (1971). "The Traveling Salesman Problem and Minimum Spanning Trees: Part II," Mathematical Programming, Vol. 1, pp. 6-25.

10. Parker, R. G. and R. L. Rardin (1982). "An Overview of Complexity Theory in Discrete Optimization: Part I. Concepts," IIE Transactions, Vol. 14, No. 1, pp. 3-10.

11. Parker, R. G. and R. L. Rardin (1982). "An Overview of Complexity Theory in Discrete Optimization: Part II. Results and Implications," IIE Transactions, Vol. 14, No. 2, pp. 83-89.

12. Rizo-Patron, A., Bozer, Y. A., and L. F. McGinnis (1983). "Analytic and Simulation Models for Advanced Automated Storage/Retrieval Systems," MHRC-TR-82-04, Material Handling Research Center, Georgia Institute of Technology, Atlanta, GA.

13. Schwarz, L. B., S. C. Graves, and W. H. Hausman (1977). "Scheduling Policies for Automatic Warehousing Systems: Simulation Results," AIIE Transactions, Vol. 10, No. 3, pp. 260-270.

Table 1. Expected Value of the Smallest of n Random Distances

	$E(Z_n)$				
n	b=.6	b=.7	b=.8	b=.9	b=1.0
1	0.3861	0.4036	0.4229	0.4440	0.4667
2	0.2703	0.2877	0.3052	0.3226	0.3397
3	0.2188	0.2343	0.2494	0.2639	0.2781
4	0.1883	0.2021	0.2154	0.2281	0.2403
5	0.1675	0.1801	0.1920	0.2034	0.2143
6	0.1522	0.1638	0.1747	0-.1851	0.1951
7	0.1404	0.1511	0.1613	0.1709	0.1801
8	0.1309	0.1410	0.1505	0.1594	0.1680
9	0.1231	0.1326	0.1415	0.1500	0.1580
10	0.1164	0.1255	0.1339	0.1420	0.1496
11	0.1108	0.1194	0.1274	0.1351	0.1424
12	0.1058	0.1141	0.1218	0.1291	0.1361
13	0.1015	0.1094	0.1168	0.1238	0.1305
14	0.0976	0.1052	0.1124	0.1191	0.1256
15	0.0942	0.1015	0.1084	0.1149	0.1211
16	0.0910	0.0982	0.1048	0.1111	0.1171
17	0.0882	0.0951	0.1016	0.1077	0.1135
18	0.0856	0.0923	0.0986	0.1045	0.1102
19	0.0832	0.0898	0.0959	0.1016	0.1071
20	0.0810	0.0874	0.0933	0.0990	0.1043
21	0.0790	0.0852	0.0910	0.0965	0.1017
22	0.0771	0.0832	0.0888	0.0942	0.0993
23	0.0753	0.0813	0.0868	0.0920	0.0970
24	0.0737	0.0795	0.0849	0.0900	0.0949
25	0.0721	0.0778	0.0831	0.0881	0.0929
26	0.0707	0.0762	0.0814	0.0864	0.0910
27	0.0693	0.0748	0.0799	0.0847	0.0893
28	0.0680	0.0734	0.0784	0.0831	0.0876
29	0.0668	0.0720	0.0770	0.0816	0.0860
30	0.0650	0.0708	0.0756	0.0802	0.0845
31	0.0645	0.0696	0.0744	0.0788	0.0831
32	0.0634	0.0685	0.0731	0.0776	0.0817
33	0.0624	0.0674	0.0720	0.0763	0.0804
34	0.0615	0.0663	0.0709	0.0752	0.0792
35	0.0606	0.0654	0.0698	0.0740	0.0780
36	0.0597	0.0644	0.0688	0.0730	0.0769
37	0.0588	0.0635	0.0678	0.0719	0.0758
38	0.0580	0.0626	0.0669	0.0710	0.0748
39	0.0573	0.0618	0.0660	0.0700	0.0738
40	0.0565	0.0610	0.0652	0.0691	0.0728

Table 2-1, $\hat{E}(TB^{NN}_{n,m})$, Average Travel-Between by Nearest-Neighbor.

n \ m	1	2	3	4	5	6	7	8	9	10
1	.4667	.3397	.2781	.2403	.2143	.1951	.1801	.1680	.1580	.1496
2	.4032	.3089	.2592	.2273	.2047	.1876	.1741	.1630	.1538	.1460
3	.3615	.2860	.2442	.2166	.1965	.1811	.1687	.1585	.1500	.1427
4	.3312	.2681	.2320	.2075	.1894	.1753	.1639	.1545	.1465	.1397
5	.3078	.2535	.2216	.1996	.1831	.1702	.1596	.1508	.1433	.1368
6	.2890	.2413	.2127	.1926	.1775	.1655	.1557	.1474	.1404	.1342
7	.2735	.2308	.2048	.1865	.1725	.1613	.1521	.1443	.1376	.1318
8	.2603	.2217	.1979	.1810	.1680	.1575	.1488	.1414	.1351	.1295
9	.2489	.2137	.1918	.1760	.1638	.1539	.1457	.1387	.1327	.1273
10	.2390	.2066	.1862	.1714	.1600	.1507	.1429	.1362	.1304	.1253
11	.2302	.2002	.1811	.1673	.1564	.1476	.1402	.1338	.1283	.1234
12	.2224	.1944	.1765	.1634	.1532	.1448	.1377	.1316	.1263	.1216
13	.2153	.1891	.1722	.1599	.1501	.1421	.1353	.1295	.1244	.1199
14	.2089	.1842	.1683	.1566	.1473	.1396	.1331	.1275	.1226	.1183
15	.2030	.1797	.1647	.1535	.1446	.1372	.1310	.1256	.1209	.1167
16	.1977	.1756	.1613	.1506	.1421	.1350	.1290	.1238	.1193	.1152
17	.1927	.1717	.1581	.1478	.1397	.1329	.1272	.1221	.1177	.1138
18	.1881	.1682	.1551	.1453	.1374	.1309	.1254	.1205	.1162	.1124
19	.1839	.1648	.1523	.1429	.1353	.1290	.1237	.1190	.1148	.1111
20	.1799	.1616	.1496	.1406	.1333	.1272	.1220	.1175	.1135	.1099
21	.1762	.1587	.1471	.1384	.1314	.1255	.1205	.1161	.1122	.1087
22	.1727	.1559	.1447	.1363	.1295	.1239	.1190	.1147	.1109	.1075
23	.1694	.1532	.1425	.1344	.1278	.1223	.1175	.1134	.1097	.1064
24	.1663	.1507	.1403	.1325	.1261	.1208	.1162	.1121	.1085	.1053
25	.1633	.1483	.1383	.1307	.1245	.1193	.1148	.1109	.1074	.1042

Table 2-2, $\hat{E}(TB^{NN}_{n,m})$, Dual Cycle Time, Nearest-Neighbor Heuristic

n \ m	1	2	3	4	5	6	7	8	9	10
1	1.7997	1.6727	1.6111	1.5733	1.5473	1.5281	1.5131	1.5010	1.4910	1.4826
2	1.7362	1.6419	1.5922	1.5603	1.5377	1.5206	1.5071	1.4960	1.4868	1.4790
3	1.6945	1.6190	1.5772	1.5496	1.5295	1.5141	1.5017	1.4915	1.4830	1.4757
4	1.6642	1.6011	1.5650	1.5405	1.5224	1.5083	1.4969	1.4875	1.4795	1.4727
5	1.6408	1.5865	1.5546	1.5326	1.5161	1.5032	1.4926	1.4838	1.4763	1.4698
6	1.6220	1.5743	1.5457	1.5256	1.5105	1.4985	1.4887	1.4804	1.4734	1.4672
7	1.6065	1.5638	1.5378	1.5195	1.5055	1.4943	1.4851	1.4773	1.4706	1.4648
8	1.5933	1.5547	1.5309	1.5140	1.5010	1.4905	1.4818	1.4744	1.4681	1.4625
9	1.5819	1.5467	1.5248	1.5090	1.4968	1.4869	1.4787	1.4717	1.4657	1.4603
10	1.5720	1.5396	1.5192	1.5044	1.4930	1.4837	1.4759	1.4692	1.4634	1.4583
11	1.5632	1.5332	1.5141	1.5003	1.4894	1.4806	1.4732	1.4668	1.4613	1.4564
12	1.5554	1.5274	1.5095	1.4964	1.4862	1.4778	1.4707	1.4646	1.4593	1.4546
13	1.5483	1.5221	1.5052	1.4929	1.4831	1.4751	1.4683	1.4625	1.4574	1.4529
14	1.5419	1.5172	1:5013	1.4896	1.4803	1.4726	1.4661	1.4605	1.4556	1.4513
15	1.5360	1.5127	1.4977	1.4865	1.4776	1.4702	1.4640	1.4586	1.4539	1.4497
16	1.5307	1.5086	1.4943	1.4836	1.4751	1.4680	1.4620	1.4568	1.4523	1.4482
17	1.5257	1.5047	1.4911	1.4808	1.4727	1.4659	1.4602	1.4551	1.4507	1.4468
18	1.5211	1.5012	1.4881	1.4783	1.4704	1.4639	1.4584	1.4535	1.4492	1.4454
19	1.5169	1.4978	1.4853	1.4759	1.4683	1.4620	1.4567	1.4520	1.4478	1.4441
20	1.5129	1.4946	1.4826	1.4736	1.4663	1.4602	1.4550	1.4505	1.4465	1.4429
21	1.5092	1.4917	1.4801	1.4714	1.4644	1.4585	1.4535	1.4491	1.4452	1.4417
22	1.5057	1.4889	1.4777	1.4693	1.4625	1.4569	1.4520	1.4477	1.4439	1.4405
23	1.5024	1.4862	1.4755	1.4674	1.4608	1.4553	1.4505	1.4464	1.4427	1.4394
24	1.4993	1.4837	1.4733	1.4655	1.4591	1.4538	1.4492	1.4451	1.4415	1.4383
25	1.4963	1.4813	1.4713	1.4637	1.4575	1.4523	1.4478	1.4439	1.4404	1.4372

Table 3. Results of Monte Carlo Sampling.

m \ n	1		3		6	
	$\hat{E}(TB^{NN}_{n,m})$	$\hat{E}(DC^{NN}_{n,m})$	$\hat{E}(TB^{NN}_{n,m})$	$\hat{E}(DC^{NN}_{n,m})$	$\hat{E}(TB^{NN}_{n,m})$	$\hat{E}(DC^{NN}_{n,m})$
1	.4581	1.8016	.2923	1.6051	.2019	1.5588
2	.4093	1.7350	.2684	1.5945	.1902	1.5060
3	.3706	1.7090	.2509	1.5678	.1874	1.5218
4	.3433	1.6729	.2408	1.5465	.1805	1.5046
5	.3169	1.6407	.2269	1.5489	.1750	1.4839
6	.2971	1.6283	.2199	1.5421	.1706	1.5100
7	.2832	1.6088	.2104	1.5398	.1634	1.4745
8	.2681	1.6026	.2040	1.5202	.1608	1.4889
9	.2571	1.5911	.1970	1.5278	.1608	1.4889
10	.2464	1.5740	0.0000	0.0000	0.0000	0.0000

Table 4. $E(DC_{n,m}^{LB})$, Lower Bound of Average Dual Cycle Time.

n \ m	1	2	3	4	5	6	7	8	9	10
1	1.7449	1.5976	1.5241	1.4787	1.4477	1.4252	1.4081	1.3948	1.3843	1.3759
2	1.6328	1.5496	1.4982	1.4630	1.4374	1.4181	1.4030	1.3911	1.3816	1.3738
3	1.5784	1.5201	1.4803	1.4513	1.4294	1.4124	1.3989	1.3880	1.3792	1.3720
4	1.5451	1.4999	1.4671	1.4422	1.4229	1.4076	1.3954	1.3854	1.3772	1.3704
5	1.5221	1.4849	1.4568	1.4349	1.4176	1.4037	1.3924	1.3831	1.3754	1.3690
6	1.5052	1.4733	1.4485	1.4288	1.4131	1.4003	1.3897	1.3810	1.3738	1.3677
7	1.4919	1.4638	1.4416	1.4237	1.4092	1.3973	1.3874	1.3792	1.3723	1.3665
8	1.4813	1.4561	1.4358	1.4194	1.4058	1.3946	1.3854	1.3776	1.3710	1.3655
9	1.4724	1.4495	1.4309	1.4155	1.4029	1.3923	1.3835	1.3761	1.3698	1.3646
10	1.4650	1.4439	1.4265	1.4122	1.4002	1.3902	1.3818	1.3748	1.3688	1.3637
11	1.4587	1.4390	1.4227	1.4092	1.3978	1.3883	1.3803	1.3735	1.3678	1.3629
12	1.4531	1.4346	1.4193	1.4065	1.3957	1.3866	1.3789	1.3724	1.3669	1.3621
13	1.4482	1.4308	1.4162	1.4040	1.3938	1.3851	1.3776	1.3714	1.3660	1.3614
14	1.4438	1.4273	1.4135	1.4018	1.3920	1.3836	1.3765	1.3704	1.3652	1.3608
15	1.4399	1.4242	1.4110	1.3998	1.3903	1.3823	1.3754	1.3695	1.3645	1.3602
16	1.4363	1.4214	1.4087	1.3980	1.3888	1.3810	1.3744	1.3687	1.3638	1.3596
17	1.4332	1.4188	1.4066	1.3963	1.3874	1.3799	1.3734	1.3679	1.3632	1.3591
18	1.4303	1.4164	1.4047	1.3947	1.3862	1.3788	1.3726	1.3672	1.3625	1.3586
19	1.4275	1.4142	1.4029	1.3932	1.3849	1.3778	1.3717	1.3665	1.3620	1.3581
20	1.4250	1.4122	1.4012	1.3918	1.3838	1.3769	1.3709	1.3659	1.3614	1.3576
21	1.4227	1.4103	1.3997	1.3906	1.3827	1.3760	1.3702	1.3652	1.3609	1.3572
22	1.4206	1.4085	1.3982	1.3893	1.3817	1.3752	1.3695	1.3646	1.3604	1.3568
23	1.4186	1.4069	1.3968	1.3882	1.3808	1.3744	1.3689	1.3641	1.3600	1.3564
24	1.4167	1.4054	1.3955	1.3872	1.3799	1.3736	1.3682	1.3636	1.3595	1.3560
25	1.4150	1.4039	1.3944	1.3861	1.3790	1.3729	1.3676	1.3631	1.3591	1.3557

Table 5. Efficiency of Nearest-Neighbor Heuristic.

n \ m	1	2	3	4	5	6	7	8	9	10
1	.0314	.0470	.0571	.0639	.0688	.0722	.0746	.0761	.0771	.0776
2	.0633	.0596	.0628	.0665	.0698	.0723	.0741	.0754	.0762	.0766
3	.0735	.0651	.0655	.0677	.0700	.0720	.0735	.0746	.0752	.0756
4	.0771	.0675	.0667	.0681	.0699	.0715	.0728	.0737	.0743	.0746
5	.0780	.0684	.0671	.0681	.0695	.0709	.0720	.0728	.0734	.0737
6	.0776	.0686	.0671	.0677	.0690	.0702	.0712	.0720	.0725	.0728
7	.0768	.0683	.0668	.0673	.0683	.0694	.0704	.0711	.0716	.0719
8	.0756	.0678	.0662	.0667	.0676	.0687	.0696	.0703	.0708	.0710
9	.0744	.0670	.0656	.0660	.0670	.0679	.0688	.0695	.0699	.0702
10	.0730	.0663	.0650	.0654	.0662	.0672	.0681	.0687	.0691	.0694
11	.0717	.0654	.0643	.0646	.0655	.0665	.0673	.0679	.0684	.0686
12	.0704	.0646	.0635	.0640	.0648	.0657	.0665	.0672	.0676	.0679
13	.0691	.0638	.0629	.0633	.0641	.0650	.0658	.0665	.0669	.0672
14	.0679	.0630	.0621	.0626	.0634	.0643	.0651	.0657	.0662	.0665
15	.0668	.0622	.0614	.0619	.0628	.0636	.0644	.0651	.0655	.0658
16	.0657	.0613	.0607	.0612	.0621	.0630	.0638	.0644	.0649	.0652
17	.0646	.0606	.0600	.0606	.0614	.0623	.0631	.0638	.0642	.0645
18	.0635	.0598	.0594	.0599	.0608	.0617	.0625	.0632	.0636	.0639
19	.0626	.0591	.0587	.0593	.0602	.0611	.0619	.0626	.0630	.0633
20	.0616	.0584	.0581	.0587	.0596	.0605	.0613	.0620	.0624	.0628
21	.0607	.0577	.0575	.0581	.0590	.0600	.0607	.0614	.0619	.0622
22	.0599	.0570	.0569	.0576	.0585	.0594	.0602	.0609	.0613	.0617
23	.0591	.0564	.0563	.0570	.0579	.0589	.0597	.0603	.0608	.0612
24	.0583	.0558	.0557	.0565	.0574	.0583	.0592	.0598	.0603	.0607
25	.0575	.0552	.0552	.0559	.0569	.0578	.0586	.0593	.0598	.0602

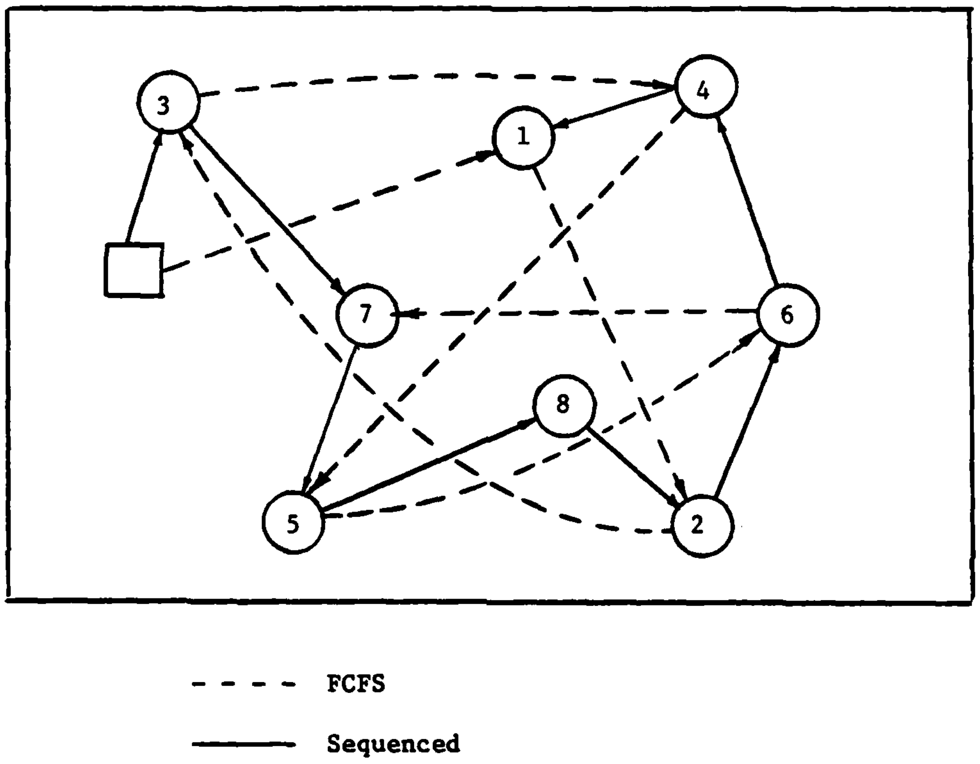


Figure 1. Example of Retrieval Sequencing.

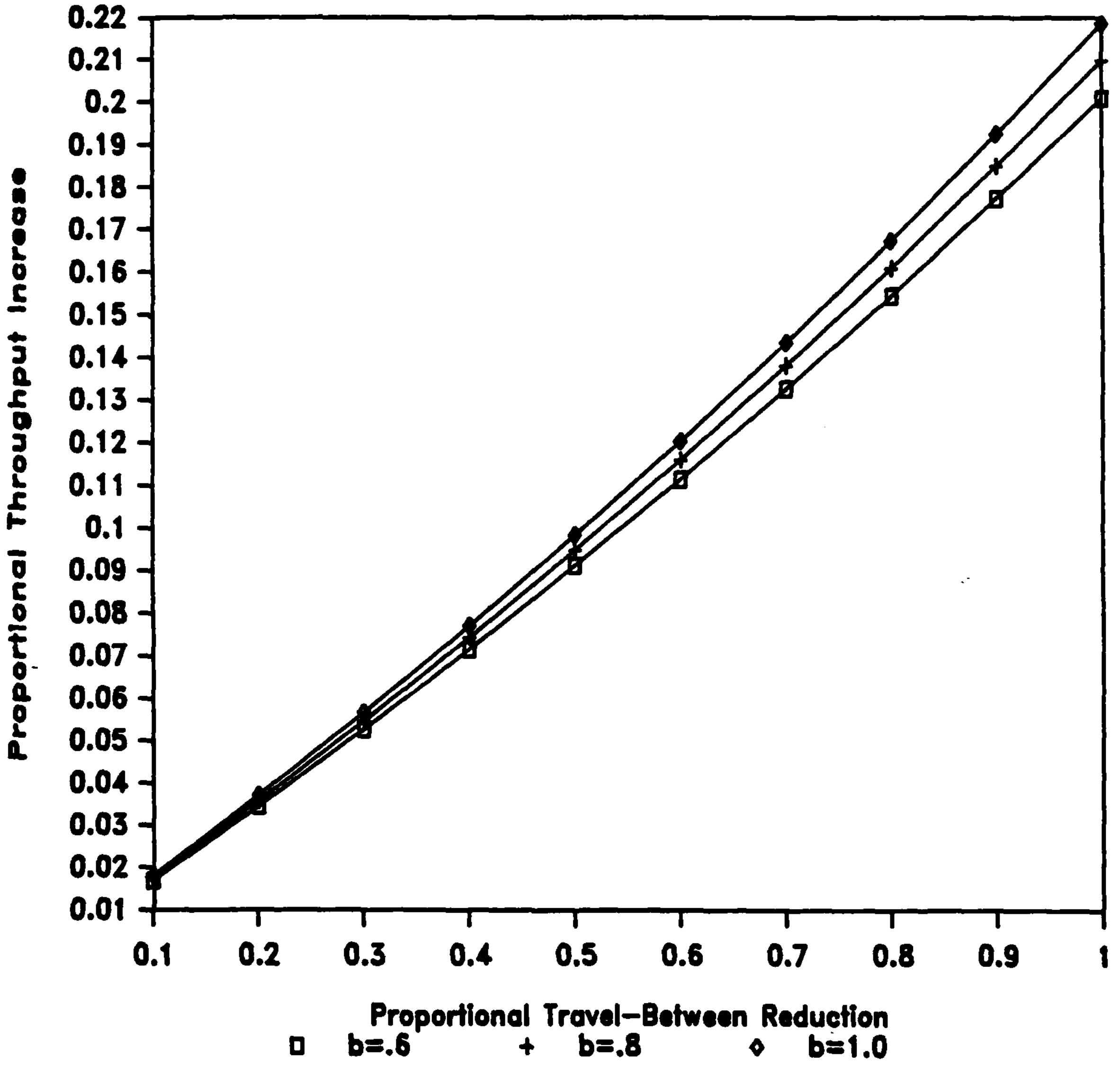


Figure 2. Throughput Improvement vs. Travel-Between Reduction.

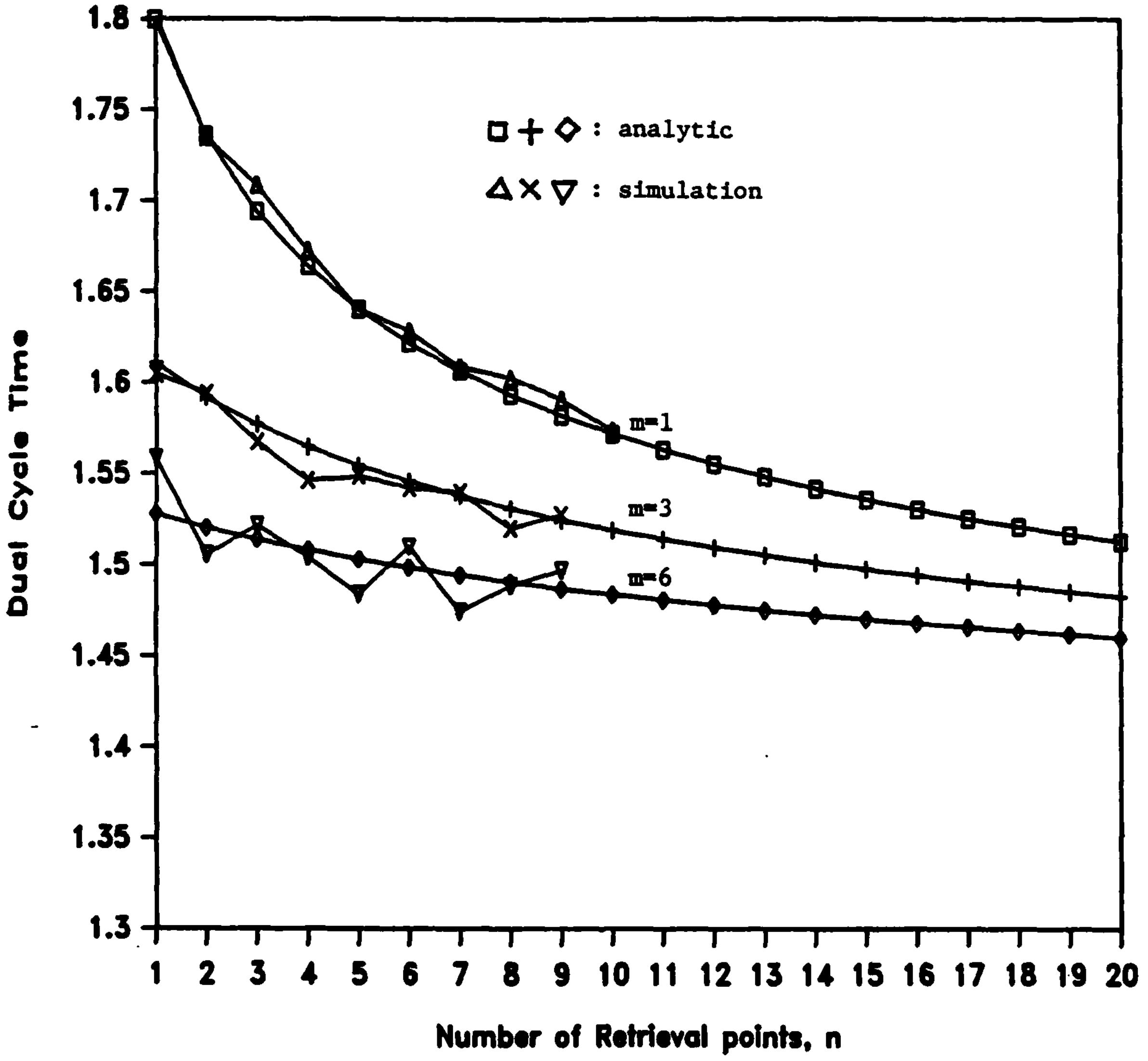


Figure 3. Expected Dual Cycle Time.

Analytic vs. Simulation

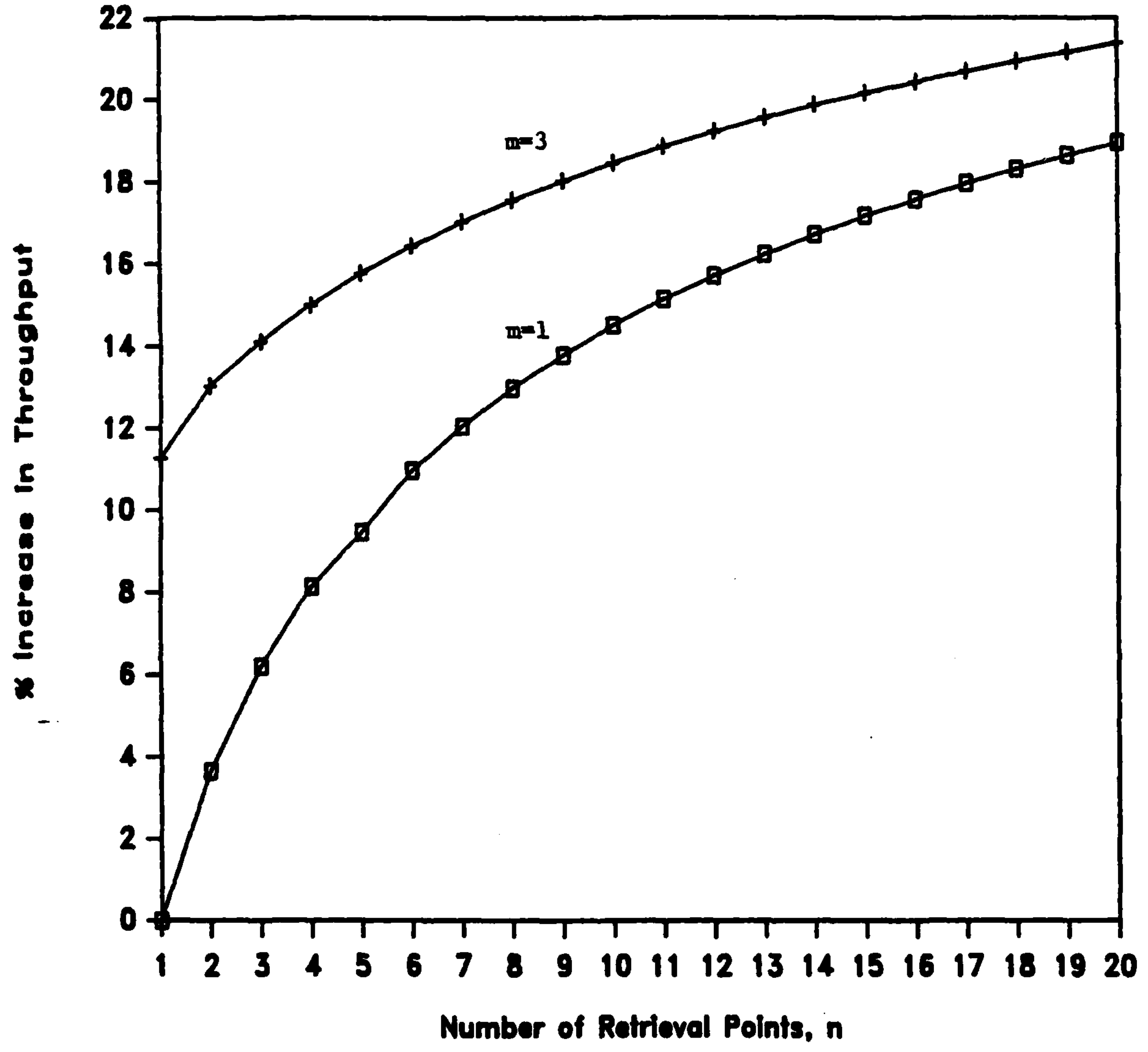


Figure 4. Throughput Improvement

Over FCFS

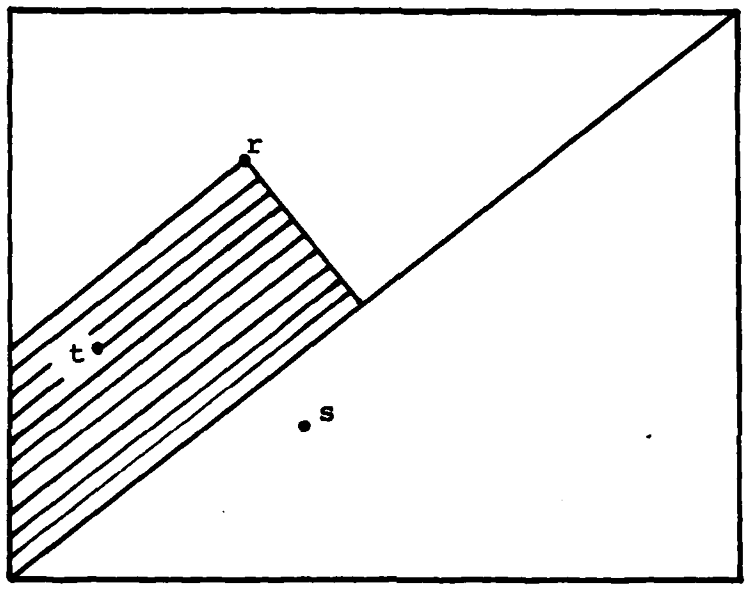


Figure 5. Illustration of No Cost Zone.

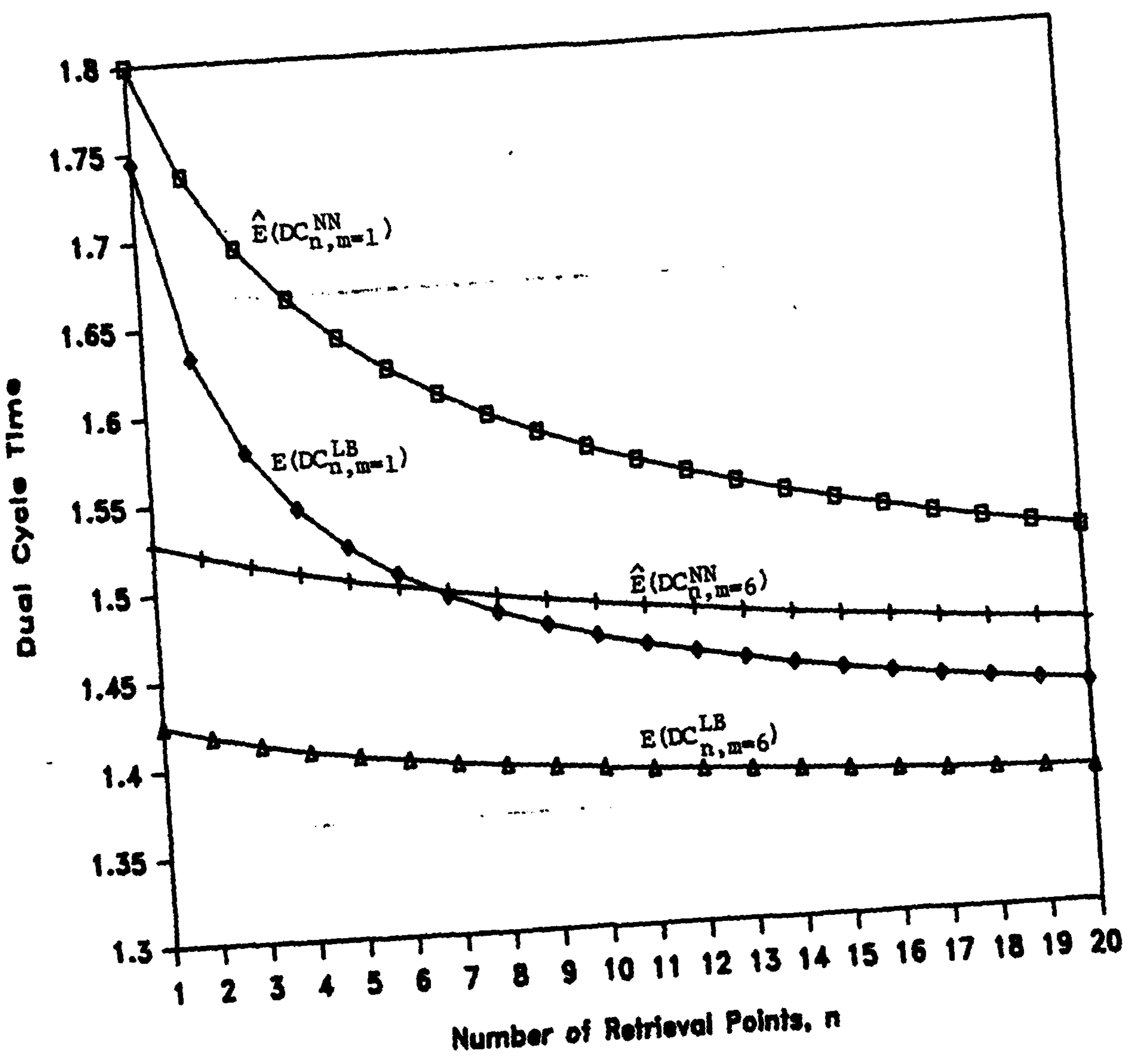


Figure 6. Performance of Nearest-Neighbor vs. Lower Bound

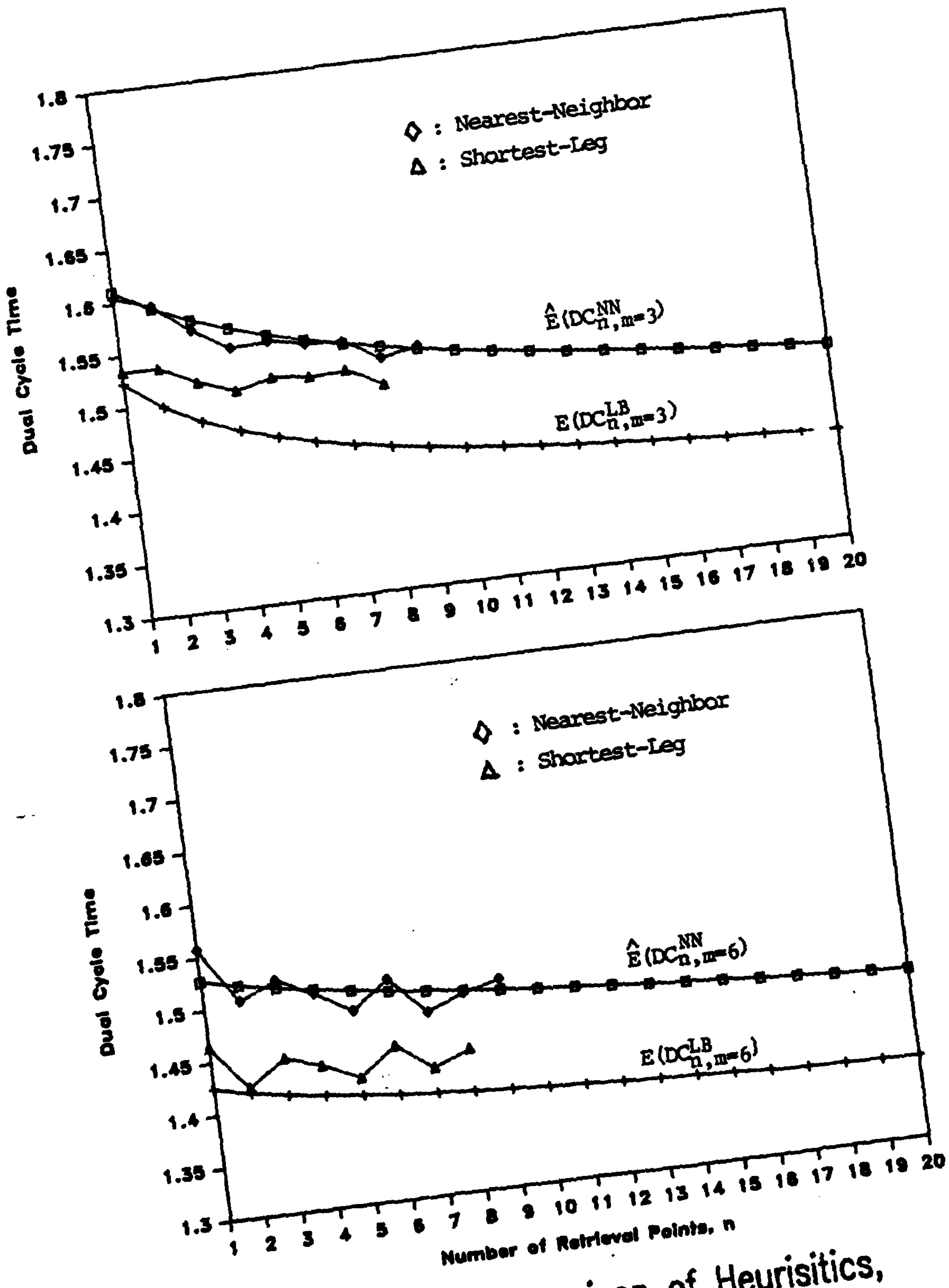
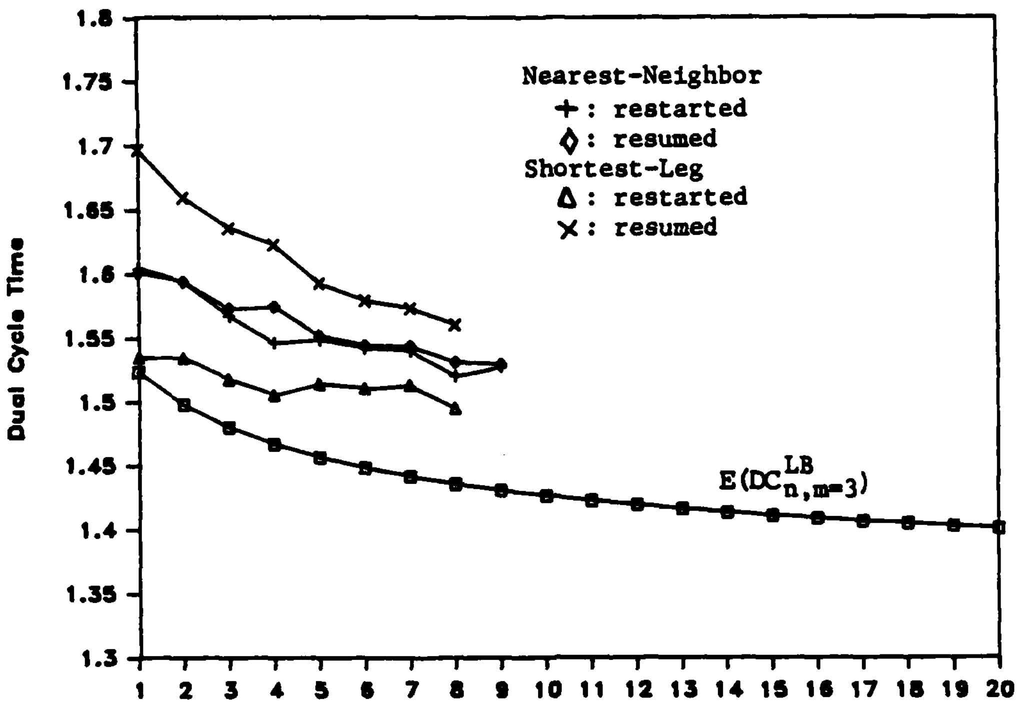


Figure 7. Comparison of Heurisitics,

Nearest-Neighbor vs. Shortest-Leg



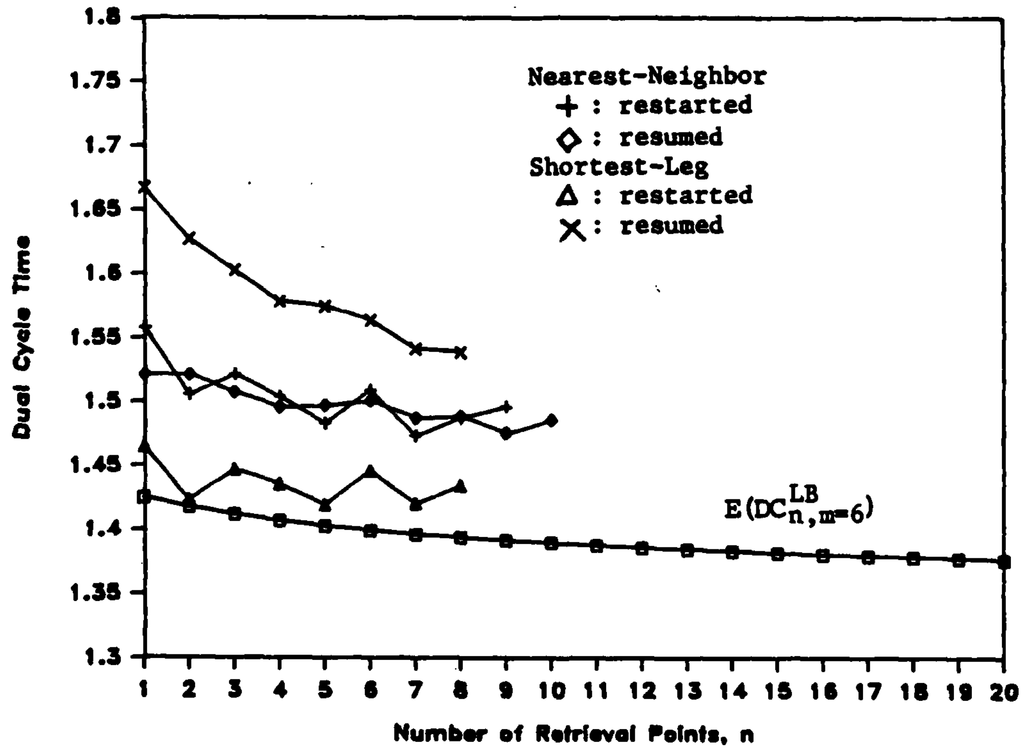


Figure 8. Dynamic Behavior,

Nearest-Neighbor vs. Shortest-Leg

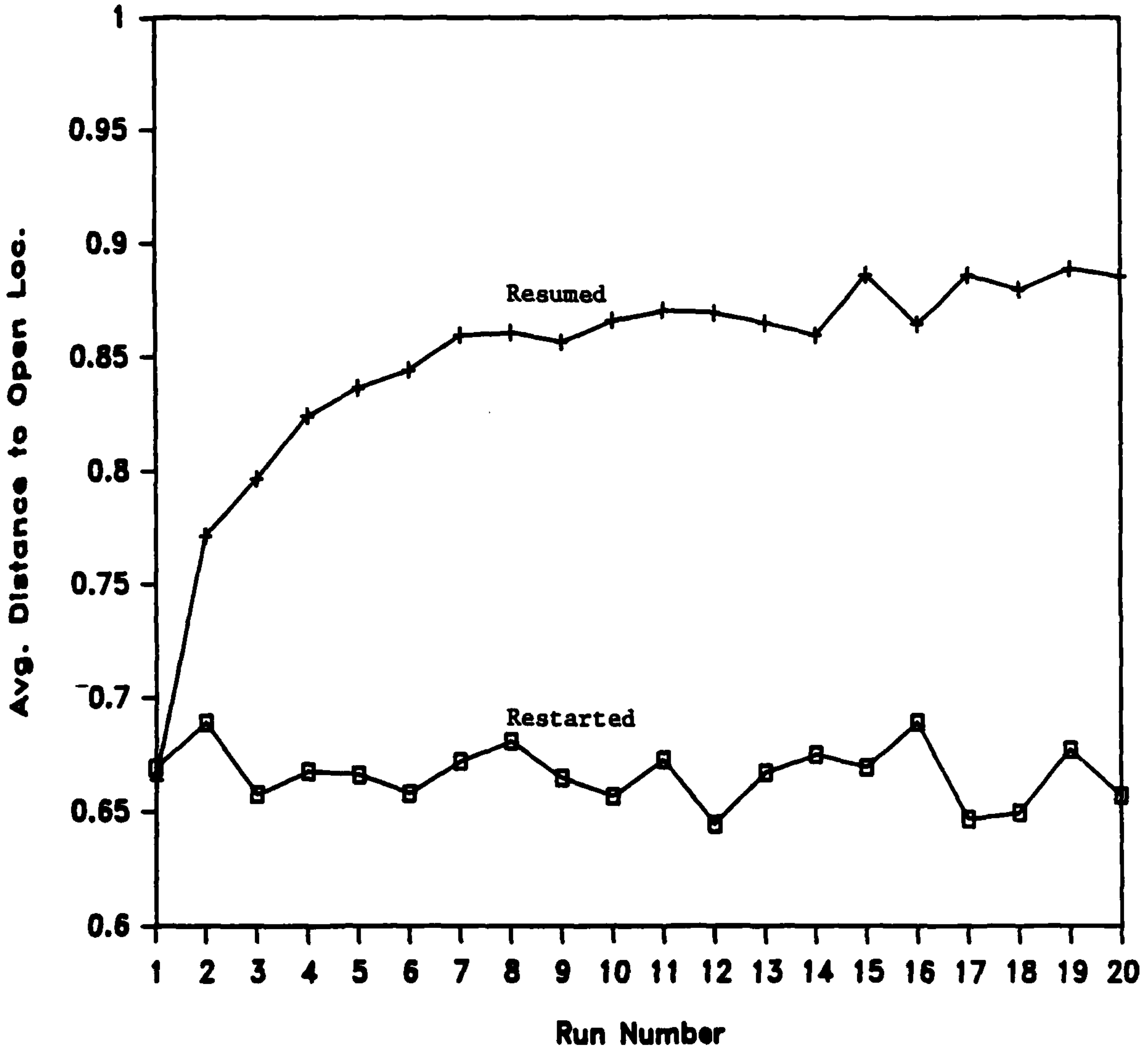


Figure 9. Migration of Open Locations,

under Shortest-Leg, n=3, m=6, sample=50